# Pandaqi Tutorials

## Migrate content

**CODE**

FIND: `(.+)`\{.longsyntax\}

REPLACE:

{{< highlight Latex >}}

$1

{{< /highlight >}}

**LONGER CODE:**

Can’t do it at once. I *can* remove the upper and lower part separately.

UPPER

FIND: “::: and stuff after it”, use [\n+] for newlines

REPLACE: {{< double-view src=”TO DO” lang=”whatever I need now” >}}

LOWER

FIND: “stuff before it :::”

REPLACE: {{< /double-view >}} $1 (if we select the image URL as well here, haven’t found a way yet)

Transfer all content

* Create new folder with useful name, add **index.md** file
* Copy the YAML stuff into frontmatter
* Transfer the post into the content side.
* Use an HTML to Markdown converter for the rest.
  + <https://codebeautify.org/html-to-markdown>
* Replace CODE BLOCK with HUGO SHORTCODE. (Must be done after HTML conversion, because it won’t recognize the shortcode as valid HTML.)
  + Find: “<pre data-lang=\\"(.+)\\">([\s\S\n]+?)</pre>” (RegExp on)
  + Replace:   
    {{< highlight $1 >}}  
    $2  
    {{< /highlight >}}
* Move all images to be within their own folder.
  + **Modify how Hugo inserts images** to use <figure>**,** like Idid before
  + **Create another shortcode for inserting video**, check Pandaqi blog
* Misc:
  + “NOTE: “ and “NOTE 2: “ need to be replaced with a *remark shortcode*
  + Replace “\>” with “>”?
  + Replace any example-like things with the *example shortcode*
  + Remove the [Course] prefix almost all articles have => but **do add it back into the title?** (Must be a great way for people to find the articles through google, and for indexing as well.)

## Usage

How to add new content in the future? How is the website structured?

### New course

* Create folder
* Add “\_index.md”
* Frontmatter: title, type: “course”, and a style
* Content: an introduction/description
* Place icon with name “icon.svg” in that folder

Once in a while, check the *Roadmap* and see if it needs updating

### New tutorial

* Create folder inside course.
* Add “index.md”
* Frontmatter: title, type: “article”, and a **weight** (used for sorting, start at 1)
* Content: the actual article
* Always add proper ALT tags to images! (They show up as side notes.)
* Place all images in the same folder as the *tutorial* itself
* The first article is called **Introduction**
* The last article is called **Conclusion** (or **Conclusion: what now?**)

### Writing articles

* Use **blockquotes** (> ) for simple indentation. (Not an example or remark, but it needs to stand out, like a definition or the main point of the article.)
* Use **example and remark shortcodes** for those
* Use **highlight** shortcode (built into Hugo) for code blocks
* Use **video** shortcode for videos
* Use **double-view** shortcode for example + image.
  + Attributes: src (url to image in same folder), lang (highlight language), split (empty, even (50/50), or none (underneath each other))
* **Tables** use the basic markdown syntax. (“|” around columns, “---“ line to separate heading and content)
  + Empty **header** cells (th) are autohidden => fill them with a “\-” dash to keep them visible.

### Custom Tools

* **LaTeX:** “enableMathJax:true” and “\\( some latex math code here \\)”" for that
* **Midi:** “midi:true” and its shortcode
* **Daw:** “daw:true” and its shortcode
* **Bamboo:** “bamboo:true” and its shortcode

### Images

* *Always* use alt-tag for side-note information.
* If I want something special, use the **image** shortcode => has no inner, so only needs opening tag
  + Src = url to image
  + Alt = alt-text + side-note text
  + Class = special class (float-left, float-right, float-clear)
* If I want something to **not be clickable**, add class “no-click”
  + Otherwise, everything I clickable by default

### Quizzes

Implemented in frontmatter, under the “quiz” param.

The format is:

quiz:

  - type: TYPE

    TYPE:

    - option1

    - option2

    options:

    - option1

- option2

Each question can be reversed (asked in the other direction). The options are ordered: the answer top **TYPE:option1** is **option:option1.** The answer to **TYPE:option2** is **option:option2**.

After the correct options, add as much nonsense answers as you like.

Multiquestions are unique:

* They *cannot* be reversed.
* Every option is a *list*: **[option1, option2, option3]**
* It adds an extra key: “sentence”. This is displayed.
  + The value “$1” is replaced by the actual option
  + The value “$B” is replaced by a blank line symbol
  + In the future, I might allow arrays for the first key, and $2, $3, etc. replace the other options

There are (currently) 7 different types:

* **Terms** => gives a term and definition
* **Actions** => gives something you want to do and how to accomplish that
* **Tools** => gives a tool and what you accomplish with it
* **Objects** => gives an object and a property/properties that belong to it
* **Parts** => gives a sentence and deconstructs what each part means/represents
  1. Has extra key “sentence”, which is the actual sentence.
* **Blanks (multi)**  => gives a blank and options that might fit
* **General (multi)**  => simply displays whatever is given
* **Images =>** displays images and what they describe/highlight in text (options)

I made myself double-type the TYPE on purpose. This made the code more specific, but also requires me to be more mindful and check it.

Images can have a “sentence” and “sentence\_reverse”. This is general. Other things can use it, if they want.

## To Do

Can I easily **extend** the colored blocks the full width of the page? Will that look better?

Do I still want to restyle code blocks? (Use that grainy BG?)

More control over navigation button colors. Create “get-most-contrasting” function (maybe SASS has this by default?)

Split Critical and Style css more efficiently (and test).

Better general design and navigation:

* Really think about *how will people find and use this website?*
* Add dark mode? Allow switching?
* On mobile, permanently (softly?) show label on grid tiles
* Now it can be quite hard to use these as *documentation/reference*, due to pagination and large blocks of styling. For example, I really like the simplicity of <https://javascript.info> => small numbers, all sections on a single page in small links
* Place remarks in the side-line instead? I really like the simple look of <https://gameprogrammingpatterns.com> => text is also smaller there, while my text is HUGE.

## Future To Do

### Quiz

* Allow multiple replacements, because often I need *two* parts to change, to get a proper sentence. (<X1>, <X2>, …) (or #1, #2, #3 …)
* The formatting is now sometimes a bit clunky. But I don’t really know how to work with strong/italic/quotes consistently with dynamic content …
* “Scroll into view” with offset, methods I tried now don’t work for me somehow?

**Arrows:** Now I use a hacky 2-element trick to get the arrows. If I ever find a cleaner way (using before, or within the same element), use that. => Can’t use *before* now, as the buttons are *masked*, so the arrow would just be cut out

Because of that, with arrows + masked images, the arrow does *not* automatically color with the hover filter on masks

At the same time, it works wonderfully in all situations?

**Decoration:** Some simple decoration with faded hexagons along the background and footer and stuff?

**Side Notes:** Find a way to keep the “side-note” element positioned correctly on the biggest screen sizes

**Once CSS container units/queries are implemented (everywhere):** I can simplify and improve the hexagon grid spacing rules. Solve any issues neatly.

**Syntax Highlighting**

* Use Hugo’s built-in thing, but generate my own colors again.
* Use the grainy-blocky-texture as the background. (Like the Phaser website does.)

**Video support:**

* Add back support for YouTube videos at the top of tutorials anyway. Maybe I might even use it some day.